CyberTalents Challenges

Web Security – Escape_202 – level medium
Writer: levith4n

Description:
I made this service to search in the files and directories in the server But my secrets leaked from file at

/radnomString_flag.txt

Flag format: Flag{}

Goals:

- Terdapat fitur yang dapat mencari file dan direktori yang ada pada server
- Format file flag adalah /radnomString_flag.txt

Solutions:

- Ini adalah tampilan halaman utama dari aplikasi web target

- Terdapat tombol untuk melihat Source Code
- Saya mencoba mengkliknya dan secara otomatis mengunduh sebuah file dengan nama src.phpppppppp
- Setelah itu, saya mencoba membuka file tersebut
- Berikut adalah konten dari file tersebut:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>File Search</title>
</head>
<body>
    <h1>Search for Files</h1>
    <form method="POST" action="">
        <label for="search">File Name:</label>
        <input type="text" id="search" name="search" required>
        <br><br>
        <button type="submit">Search</button>
    </form>
    <hr>
    <?php
    if ($_SERVER["REQUEST_METHOD"] === "POST") {
        $search = escapeshellcmd($_POST['search']);


        $command = "find / -name " .$search;
```

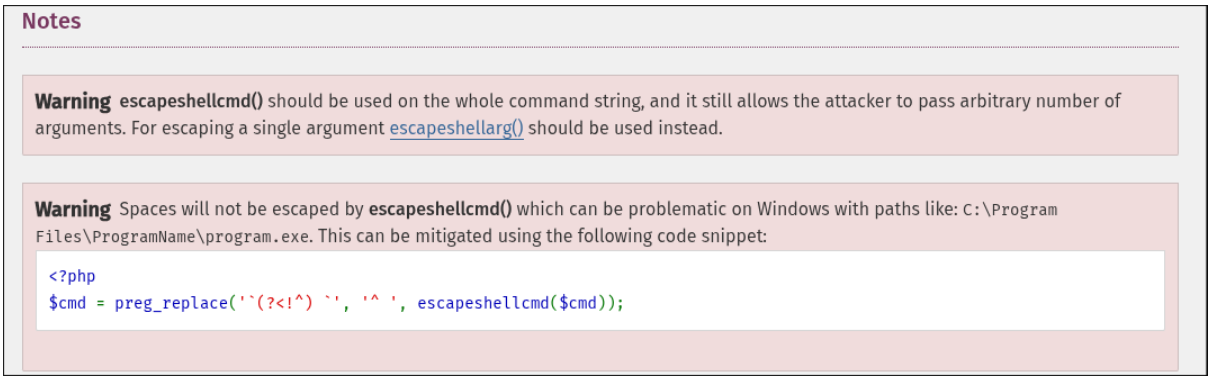
```
        $output = shell_exec($command);

        echo "<h2>Search Results:</h2>";
        if ($output) {
            echo "<pre>" . $output . "</pre>";
        } else {
            echo "<p>No files found matching your search criteria.</p>";
        }
    }

    ?>
    <a href="/src.phpppppppp" title="View the source code" download>View Source Code</a>

</body>
</html>
```

- Jika diperhatikan, terdapat kode PHP yang rentan akan command injection, yaitu pada bagian berikut:

```
if ($_SERVER["REQUEST_METHOD"] === "POST") {
        $search = escapeshellcmd($_POST['search']);


        $command = "find / -name " .$search;
        $output = shell_exec($command);

        echo "<h2>Search Results:</h2>";
        if ($output) {
            echo "<pre>" . $output . "</pre>";
        } else {
            echo "<p>No files found matching your search criteria.</p>";
        }
    }

    ?>
```

- Jadi, input pengguna yang diambil dari parameter **search** akan disanitasi terlebih dahulu menggunakan fungsi **escapeshellcmd**

- Namun, jika kita baca dokumentasi tentang **escapeshellcmd**, fungsi ini memiliki kerentanan



Notes

Warning **escapeshellcmd()** should be used on the whole command string, and it still allows the attacker to pass arbitrary number of arguments. For escaping a single argument escapeshellarg() should be used instead.

Warning Spaces will not be escaped by **escapeshellcmd()** which can be problematic on Windows with paths like: `C:\Program Files\ProgramName\program.exe`. This can be mitigated using the following code snippet:

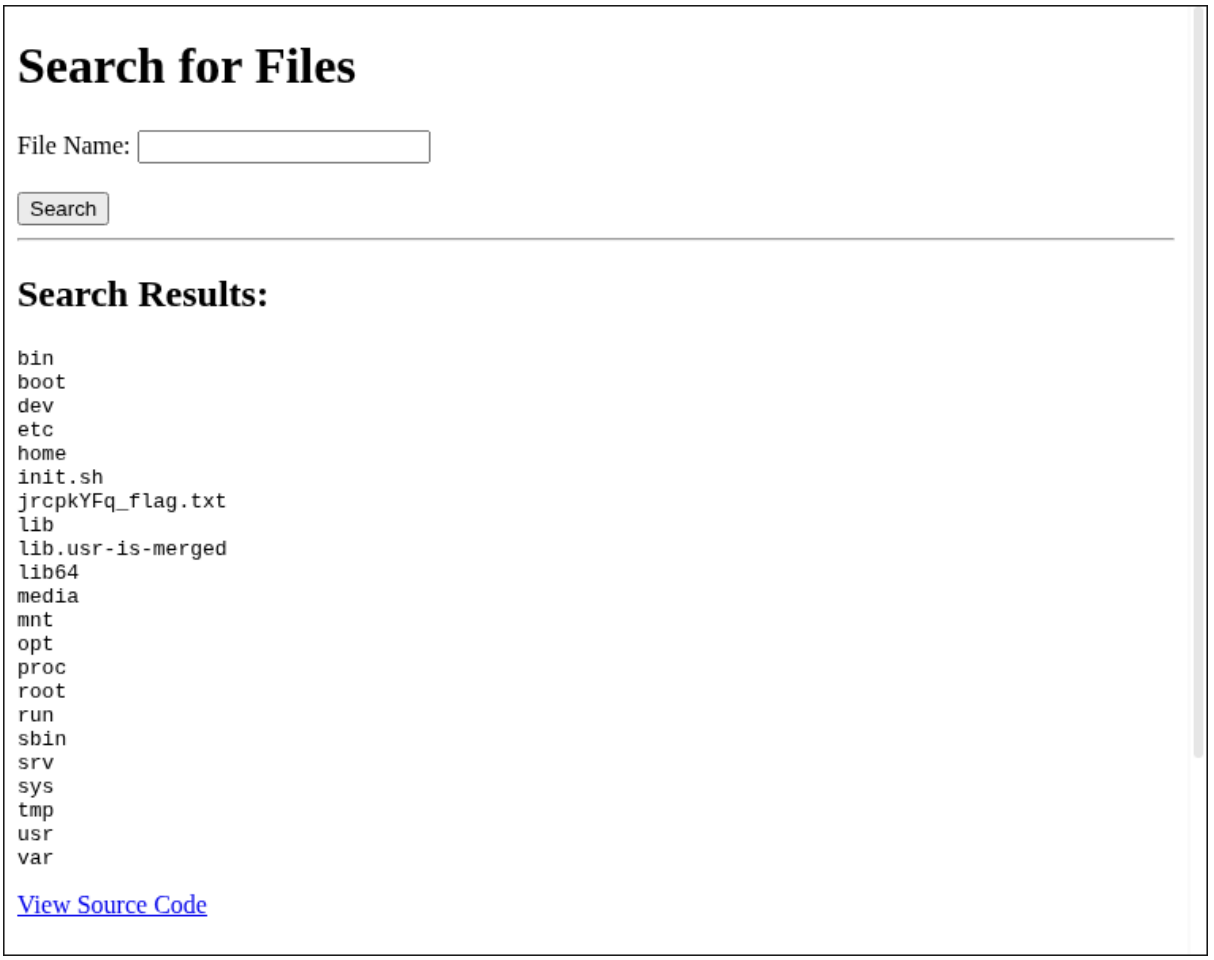
```
<?php
$cmd = preg_replace('`(?<!^) `', '^ ', escapeshellcmd($cmd));
```

- Walaupun **escapshellcmd** membersihkan input untuk tujuan menghindari command injection
- Fungsi ini tetap bisa mengizinkan penyerang melakukan command injection melalui argumen atau parameter pada perintah shell
- Pada source code, kita melihat bahwa perintah find digunakan untuk mencari file yang diinputkan oleh pengguna
- Menurut artikel yang saya dapatkan saat searching di internet, kita bisa menggunakan payload seperti berikut:

```
x -or -exec whoami ; -quit
```

- Payload ini memiliki fungsi sebagai berikut:
    - File dengan nama “x” dicari, tetapi tidak ada yang artinya ini gagal
    - Sehingga argumen setelah **-or** akan dijalankan, yaitu **-exec whoami ;**
    - Setelah satu kali pencarian maka **-quit** akan dijalankan
    - Sehingga perintah find akan langsung diberhentikan

- Setelah itu, saya menjalan perintah **ls /** dan menemukan nama file flag



- Terakhir, saya membaca konten file flag menggunakan perintah **cat**